AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A computer programming method comprising:

describing data types as abstract data types without default or implicit implementations, the abstract data types comprising only non-concrete data types;

distinguishing the abstract data types ~~and distinctly from~~ from classes or interfaces, the classes or interfaces classified as either of abstract and concrete;

~~describing subtype and supertype relationships between the types;~~ and

describing representations of values of the abstract data types as states of classes of objects with corresponding interfaces.

2. (Currently Amended) The computer programming method of claim 1 further comprising:

describing software-visible physical objects ~~and an instruction set~~of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes; and

identifying when methods of the classes are implemented directly by athe computer as instructions in an instruction set.

3. (Original) The computer programming method of claim 1 further comprising:

describing multiple classes of pointer objects, said pointer objects capable of signifying objects.

4. (Original) The computer programming method of claim 1 further comprising:

describing commands for transferring program control in non-sequential ways as routines.

5. (Currently Amended) The computer programming method of claim 1 further comprising:

describing ~~at least one of~~ interfaces, classes, enumerations, subroutines, and other routines as classes of objects.

6. (Original) The computer programming method of claim 1 further comprising:

invoking statements in a compilation with results thereof being incorporated into an output of a compiler.

7. (Original) The computer programming method of claim 1 further comprising:

invoking statements in a compilation to interpret literals in an input of a compiler.

8. (Original) The computer programming method of claim 1 further comprising:

deriving variable classes from a constant class; and

describing one of the variable classes and the constant class using a single descriptor.

9. (Original) The computer programming method of claim 1 further comprising:

deriving variable interfaces from a constant interface; and

describing one of the variable interfaces and the constant interface using a single descriptor.

10. (Currently Amended) The computer programming method of claim 1 further comprising:

describing a specification of formal arguments to routines as instances of objects, the objects' states representing the formal arguments required by the routines.

11. (Original) The computer programming method of claim 1 further comprising:

describing formal arguments to routines as a number of arguments including type, interface, or class of each argument, dataflow attribute of each argument, and preconditions and postconditions of routines.

12. (Original) The computer programming method of claim 1 further comprising:

describing subroutines as parameterized classes.

13. (Currently Amended) A method of compilation comprising:

a. generating a description of a computer architecture as a first library, the generating a description including describing software-visible physical objects ~~and the instruction set~~ of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes, and at least one of the methods is described as implemented directly by instructions in an instruction set;

b. implementing a second library of high level objects using the first library; and

c. binding a source program to implementations in the second library to produce machine instructions dependent on the computer architecture.

14. (Currently Amended) A method of re-targeting comprising:

a. generating a first description of a first computer architecture as a first library, the generating a first description including describing software-visible physical objects and an instruction set of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing

software-visible physical objects is performed using object-oriented classes, and at least one of the methods is described as implemented directly by instructions in the instruction set~~therefor~~;

b. generating a second description of a second computer architecture as a second library, the second description including software-visible objects and instruction set therefor;

c. implementing the first description using the second library to produce a third library; and

d. binding a source program to implementations in the third library to produce machine instructions dependent on the second computer architecture.

15. (Currently Amended) A computer programming method comprising:

describing software-visible physical objects ~~and an instruction set~~of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes; and

identifying when methods of the classes are implemented directly by ~~a~~the computer as instructions in an instruction set.

16. (Original) A computer programming method comprising:

deriving variable classes from a constant class; and

describing one of the variable classes and the constant class using a single descriptor.

17. (Currently Amended) A storage medium encoded with machine-readable code, the code including instructions for allowing a computer to implement a computer programming method comprising:

describing data types as abstract data types without default or implicit implementations, the abstract data types comprising only non-concrete data types;

distinguishing the abstract data types ~~and distinctly~~ from classes or interfaces, the classes or interfaces classified as either of abstract and concrete;

~~describing subtype and supertype relationships between the types;~~ and

describing representations of values of the abstract data types as states of classes of objects with corresponding interfaces.

18. (Currently Amended) The storage medium of claim 17 wherein the method further comprises:

describing software-visible physical objects ~~and an instruction set~~ of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes; and

identifying when methods of the classes are implemented directly by ~~a~~the computer as instructions in an instruction set.

19. (Original) The storage medium of claim 17 wherein the method further comprises:

describing multiple classes of pointer objects, said pointer objects capable of signifying objects.

20. (Original) The storage medium of claim 17 wherein the method further comprises:

describing commands for transferring program control in non-sequential ways as routines.

21. (Currently Amended) The storage medium of claim 17 wherein the method further comprises:

describing ~~at least one of~~ interfaces, classes, enumerations, subroutines, and other routines as classes of objects.

22. (Original) The storage medium of claim 17 wherein the method further comprises:

invoking statements in a compilation with results thereof being incorporated into an output of a compiler.

23. (Original) The storage medium of claim 17 wherein the method further comprises:

invoking statements in a compilation to interpret literals in an input of a compiler.

24. (Original) The storage medium of claim 17 wherein the method further comprises:

deriving variable classes from a constant class; and

describing one of the variable classes and the constant class using a single descriptor.

25. (Original) The storage medium of claim 17 wherein the method further comprises:

deriving variable interfaces from a constant interface; and

describing one of the variable interfaces and the constant interface using a single descriptor.

26. (Currently Amended) The storage medium of claim 17 wherein the method further comprises:

describing a specification of formal arguments to routines as instances of objects, the objects' states representing the formal arguments required by the routines.

27. (Original) The storage medium of claim 17 wherein the method further comprises:

describing formal arguments to routines as a number of arguments including type, interface, or class of each argument, dataflow attribute of each argument, and preconditions and postconditions of routines.

28. (Original) The storage medium of claim 17 wherein the method further comprises:

describing subroutines as parameterized classes.

29. (Currently Amended) A storage medium encoded with machine-readable code for compilation, the code including instructions for causing a computer to implement a method comprising:

a. generating a description of a computer architecture as a first library, the generating athe description including describing software-visible physical objects ~~and the instruction set~~ of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes, and at least one of the methods is described as implemented directly by instructions in an instruction set;

b. implementing a second library of high level objects using the first library; and

c. binding a source program to implementations in the second library to produce machine instructions dependent on the computer architecture.

30. (Currently Amended) A storage medium encoded with machine-readable code for re-targeting, the code including instructions for causing a computer to implement a method comprising:

a. generating a first description of a first computer architecture as a first library, the generating a first description including describing software-visible physical objects and an instruction set of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes, and at least one of the methods is described as implemented directly by instructions in the instruction set ~~therefor~~;

b. generating a second description of a second computer architecture as a second library, the second description including software-visible objects and instruction set therefor;

c. implementing the first description using the second library to produce a third library; and

d. binding a source program to implementations in the third library to produce machine instructions dependent on the second computer architecture.

31. (Currently Amended) A storage medium encoded with machine-readable code, the code including instructions for allowing a computer to implement a computer programming method:

describing software-visible physical objects ~~and an instruction set~~of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes; and

identifying when methods of the classes are implemented directly by a computer as instructions in an instruction set.

32. (Original) A storage medium encoded with machine-readable code, the code including instructions for allowing a computer to implement a computer programming method:

deriving variable classes from a constant class; and

describing one of the variable classes and the constant class using a single descriptor.

33. (Currently Amended) A signal propagated over a propagation medium, the signal encoded with code including instructions for allowing a computer to implement a computer programming method comprising:

describing data types as abstract data types without default or implicit implementations, the abstract data types comprising only non-concrete data types;

distinguishing the abstract data types ~~and distinctly~~ from classes or interfaces, the classes or interfaces classified as either of abstract and concrete;

~~describing subtype and supertype relationships between the types;~~ and

describing representations of values of the abstract data types as states of classes of objects with corresponding interfaces.

34. (Currently Amended) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

describing software-visible physical objects ~~and an instruction set~~of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes; and

identifying when methods of the classes are implemented directly by a computer as instructions in an instruction set.

35. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

describing multiple classes of pointer objects, said pointer objects capable of signifying objects.

36. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

describing commands for transferring program control in non-sequential ways as routines.

37. (Currently Amended) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

describing ~~at least one of~~ interfaces, classes, enumerations, subroutines, and other routines as classes of objects.

38. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

invoking statements in a compilation with results thereof being incorporated into an output of a compiler.

39. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

invoking statements in a compilation to interpret literals in an input of a compiler.

40. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

deriving variable classes from a constant class; and

describing one of the variable classes and the constant class using a single descriptor.

41. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

deriving variable interfaces from a constant interface; and

describing one of the variable interfaces and the constant interface using a single descriptor.

42. (Currently Amended) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

describing a specification of formal arguments to routines as instances of objects, the objects' states representing the formal arguments required by the routines.

43. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

describing formal arguments to routines as a number of arguments including type, interface, or class of each argument, dataflow attribute of each argument, and preconditions and postconditions of routines.

44. (Original) The signal propagated over the propagation medium of claim 33 wherein the method further comprises:

describing subroutines as parameterized classes.

45. (Currently Amended) A signal propagated over a propagation medium, the signal encode with code for compilation, the code including instructions for causing a computer to implement a method comprising:

a. generating a description of a computer architecture as a first library, the generating a description including describing software-visible physical objects ~~and the instruction set~~ of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes, and at least one of the methods is described as implemented directly by instructions in an instruction set;

b. implementing a second library of high level objects using the first library; and

c. binding a source program to implementations in the second library to produce machine instructions dependent on the computer architecture.

46. (Currently Amended) A signal propagated over a propagation medium, the signal encode with code for re-targeting, the code including instructions for causing a computer to implement a method comprising:

a. generating a first description of a first computer architecture as a first library, the generating a first description including describing software-visible physical objects and an instruction set of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes, and at least one of the methods is described as implemented directly by instructions in the instruction set~~therefor~~;

b. generating a second description of a second computer architecture as a second library, the second description including software-visible objects and instruction set therefor;

c. implementing the first description using the second library to produce a third library; and

d. binding a source program to implementations in the third library to produce achine instructions dependent on the second computer architecture.

47. (Currently Amended) A signal propagated over a propagation medium, the signal encoded with code including instructions for allowing a computer to implement a computer programming method comprising:

describing software-visible physical objects ~~and an instruction set~~ of a computer as instances of classes, the classes comprised of methods for accessing and manipulating the software-visible physical objects, wherein the describing software-visible physical objects is performed using object-oriented classes; and

identifying when methods of the classes are implemented directly by ~~a~~the computer as instructions in an instruction set.

48. (Original) A signal propagated over a propagation medium, the signal encoded with code including instructions for allowing a computer to implement a computer programming method comprising:

deriving variable classes from a constant class; and

describing one of the variable classes and the constant class using a single descriptor.